A Novel Approach for Code Clone Detection Using Hybrid Technique

Muneer Ahmad[1], Mudasirahma Dmutto[2]

[1]RN Engineering & Management College Rohtak, Haryana, India
[2]Department of CSE, Assistant Professor, SSM College of Engineering &Technology, Jammu and Kashmir, India

Abstract— *Code clones have been studied for long, and there is strong evidence that they are a major source of software faults. The copying of code has been studied within software engineering mostly in the area of clone analysis. Software clones are regions of source code which are highly similar; these regions of similarity are called clones, clone classes, or clone pairs In this paper a hybrid approach using metric based technique with the combination of text based technique for detection and reporting of clones is proposed. The Proposed work is divided into two stages selection of potential clones and comparing of potential clones using textual comparison. The proposed technique detects exact clones on the basis of metric match and then by text match.*

Keywords— code clone, Hybrid, Textual clones, Functional clone.

I. INTRODUCTION

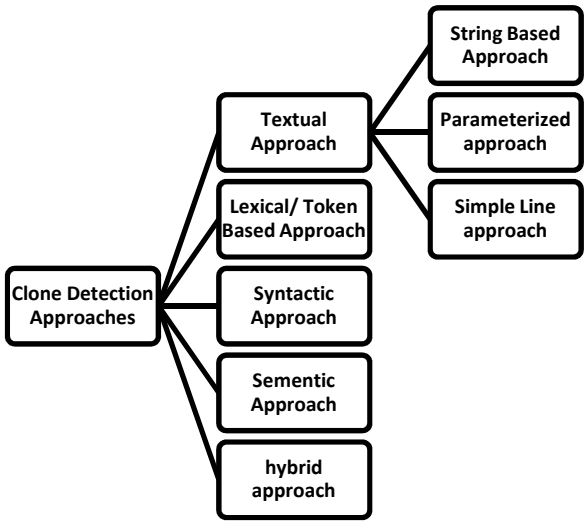
Code clones have been studied for long, and there is strong evidence that they are a major source of software faults. The copying of code has been studied within software engineering mostly in the area of clone analysis. Software clones are regions of source code which are highly similar; these regions of similarity are called clones, clone classes, or clone pairs. There are several reasons why two regions of code may be similar, the majority of the clone analysis literature attributes cloning activity to the intentional copying and duplication of code by programmers; clones may also be attributable to automatically generated code, or the constraints imposed by the use of a particular framework or library. Software clones are important aspects in software evolution. If a system is to be evolved, its clones should be known in order to make consistent changes. Cloning is often a strategic means for evolution. Recently, various clone detection techniques have been subject to empirical comparisons to compare how well they perform.

Cloning works at the cost of increasing lines of code without adding to overall productivity. It results to excessive maintenance cost. Along with these negative impacts Due to unnecessary increase in complexity and length it becomes more difficult to edit the code hence, leading to increased human errors, high maintenance cost, forgotten or overlooked codes, and increased size of the code. Also along with the duplication of code it also duplicates the errors thus increasing the errors in the code file and decreasing its efficiency.

Identifying code clones serves many purposes, including studying code evolution, performing plagiarism detection, enabling refactoring such as procedure ex-traction, and performing defect tracking and repair. Most previous work on code-clone detection has focused on finding identical clones, or clones that could be made identical via consistent transformations of identifiers and literals. However, code segments that are similar but not identical occur often in practice, and finding such non-identical clones can be as important as finding identical code segments.

The basic classification of the clones can be summed up in two categories *Textual clones* and *Functional clone.*

There are number of developers that have studied the techniques for the detection of the clones and there are number of approaches for the purpose of clone detection in any program. Few of them being Textual approach, Token-Based approach, Syntactic approach and Semantic approach.



II. CLONE DETECTION PROCESS

Clone detection process involves several steps that are:

- Pre-processing
- Transformation
- Extraction
- Normalization
- Match-detection
- Formatting
- Post-processing
- Aggregation

III. RELATED WORK

PriyankaBatta[1] Software Clone detection helps in detecting duplicate code from applications. Cloning creates problem when a bug is found in one code segment that was copied and pasted at several locations earlier. The objective of this study is to analyze the working of hybrid clone detection technique that design and analyze a hybrid technique for detecting software clone in an application. A model will be designed to automate the concept of clone detection.

Ali Selahmat and Norfaradilla Wahid [2] as the number of web pages increases across time number of clones among source code also increases. Aim is to be familiar with ontology mapping technique to solve the clone detection between files of different systems.

Florian Deissenboeck et al[3] Cloned code is considered harmful for two reasons: (1) multiple,possibly unnecessary, duplicates of code increase maintenancecosts and, (2) inconsistent changes to cloned code can create faults and, hence, lead to incorrect program behavior. Based on an industrial case study undertaken with the BMW Group, this paper details on these challenges and presents solutions to the most pressing ones, namely scalability and relevance of the results. Moreover, we present tool support that eases the evaluation of detection results and thereby helps to make clone detection a standard technique in modelbased quality assurance.

Chanchal K. Roy, et al [4] They provide a qualitative comparison and evaluation of the current state-of-the-art inclone detection techniques and tools, and organize the large amount of information into a coherentconceptual framework.

Shinji Kawaguchi,et al [5] code clones decrease the maintainability and reliability of software programs, thus it is being regarded as one of the major factors to increase development/maintenance cost.

HoanAnh Nguyen, et al [6] Structure-oriented approaches in clone detection have becomePopular in both code-based and model-based clone detection. However, existing methods for capturing structural information in software artifacts are either too computationally expensive to be efficient or too light-weight to be accurate in clone detection.

IV. OBJECTIVE AND METHODOLOGY OF THE WORK

The main objective of the study is to create a clone detection technique that is compatible with multiple languages and to propose a novel clone detection technique for Object Oriented and Platform Independent Language i.e. Java and Web application based Languages i.e. JSP (Java Server Pages), asp.net, html, PHP.

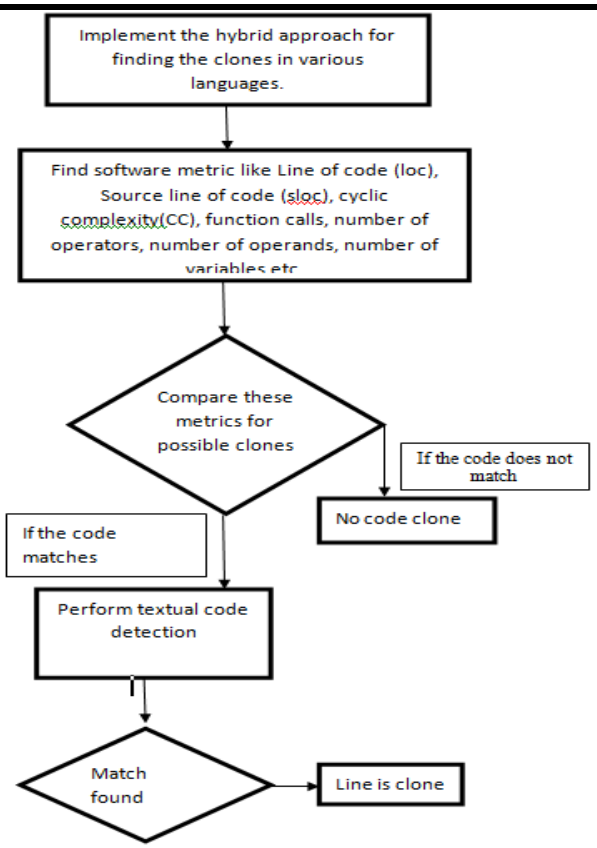
The method used to achieve this objective includes Implementation of a hybrid approach for finding the clones in various languages and then finding software metrics like line of code (loc), Source line of code (sloc), cyclic complexity (CC), function calls, number of operators, number of operands, number of variables etc. from the two files or applications and then the Comparison of these metrics for potential clones is performed; if there is match then go for textual comparison i.e. two source files are compared line by line to check for clones detection; if match found then print that line as clone. Otherwise there is no clone in the code.

Performance Metrics

Performance metrics gives the identification of potential clone in both the testing files. If there is existence of some potential clone then only we can do the textual line by line comparison of the two files. If there is no common performance metrics that means; there is no potential clone and we don't go for the textual comparison. This will save the time and gives efficient results.

There are several parameters that are used in the code clone detection techniques. Some of these parameters are LOC: No of lines in code, Public Variables, Private Variables, Protected Variables, Public Functions, Private Functions, Protected functions, If statements, Loop Statements, Redirect statements

The proposed Clone detection algorithm:

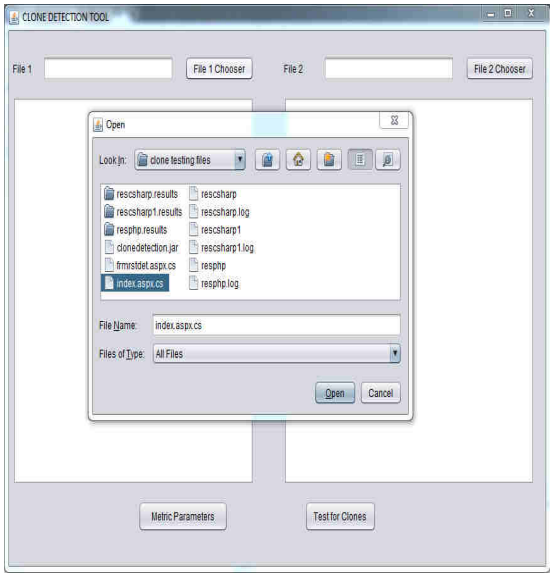


Software description:

The simulation has been done in Java Net Beans. NetBeans is an open-source software development platform dedicated to develop many other software development products, for instance, the
NetBeans IDE and the NetBeans Platform. This platform is developed to meet the needs of developers, users and the businesses that highly depend on NetBeans to develop their products quickly, efficiently and easily by taking advantage of the strengths of the Java platform and other relevant industry standards

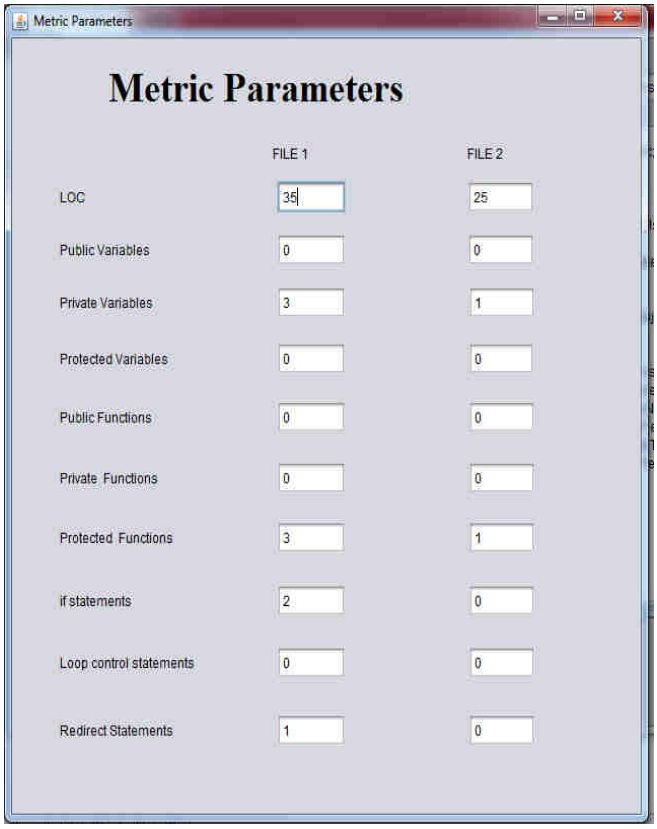
V. RESULTS

Choosing the first web application file to test for clone

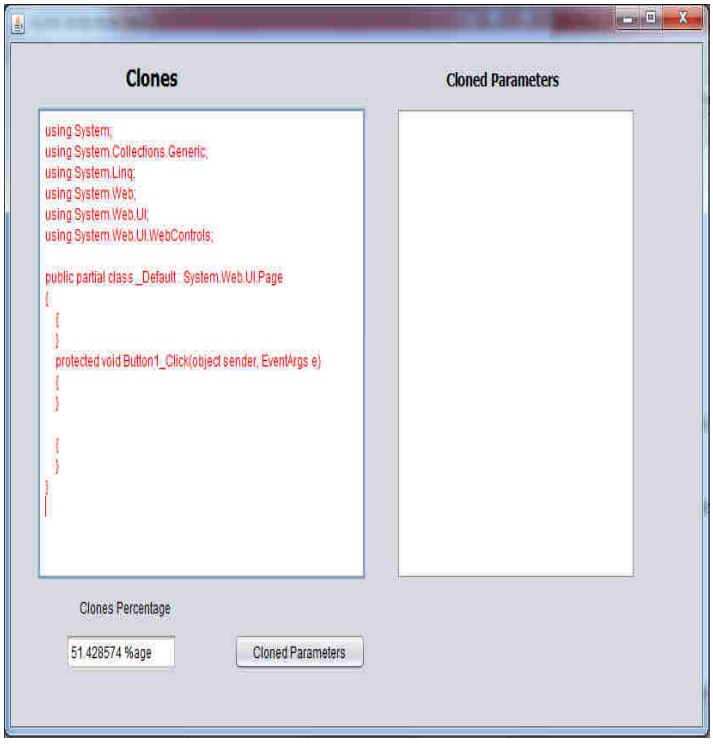


Choosing the second web application file to test for clone

The above figure shows that the 2 web based application files of asp.net are chosen. Firstly the metric parameters from both the chosen web based files are calculated which includes: LOC: No of lines in code, Public Variables, Private Variables, Protected Variables, Public Functions, Private Function, Protected functions, If statements, Loop Statements



Shows the metric parameters of both the web application aspx .csfiles in order to test for the clones



Showing the detected clones from the 2 uploaded files and the clone parameters

Detection Parameters	
Value	
Similarity Threshold	95%
Maxumum parameter count	6
Minimum Mass (Lines)	6.0
Characters per node	16
Starting height	2

Clone Detection Statistics	
Statistic	Value
File Count	2
Total Source Lines of Code (SLOC)	248
Estimated SLOC before preprocessing	238
Expanded SLOC after preprocessing	238
Total CloneSets	1
Exact-match CloneSets	1
Near-miss CloneSets	0
Number of cloned SLOC	238
SLOC in clones %	100.0%
Estimated removable SLOC	117
Possible SLOC reduction %	49.2%
Possible SLOC reduction in expanded file %	49.2%

Showing the detected clones from the 2 uploaded files and the clone percentage using clone detector tool

The above results give the comparison of our proposed technique with the existing Clone Detector tool which is also used to test the asp.net web based files. Clone detector tool has very less number of performance metrics; i.e it can't able to detect efficiently the potential clones. It is only dependent on LOC line of code and will do the textual comparison and give the clone percentage and also time consuming approach. The Proposed technique is efficient as it uses large number of performance metrics in order to find the potential clone; it there exist potential clone the only textual comparison of the files is done in order to find the clone percentage.

VI. CONCLUSION

Software clone is a phenomenon in large software system. It is usually caused by programmers' copy and paste activities. The reason for the existence of clones in the source code is that making a copy of a code fragment is simpler and faster than writing it from scratch. Sometimes maintenance programmers do not fully understand the program and therefore they re-implement some already existing functionality. Such duplications increase code size and also maintenance and comprehension becomes more difficult. Because as much as 80% of the total life cycle cost is spent on maintenance, clone elimination could translate into substantial savings.

In this thesis a hybrid approach using metric based technique with the combination of text based technique for detection and reporting of clones is proposed. The Proposed work is divided into two stages selection of potential clones and comparing of potential clones using textual comparison. The proposed technique detects exact clones on the basis of metric match and then by text match.

REFERENCES

[1] BattaPriyanka," Hybrid technique for software code clone detection" International Journal of Computer and technology, Volume 2, Issue 2, April 2012

[2] Selamatali&wahidnorfaradilla,"code clone detection using string based matching technique" International symposium on empirical software engineering" volume 2, Issue 4 April, 2005

[3] Deissenboeck Florian, Hummel Benjamin JuergensElmar, Pfaehler Michael," Model clone detection in Practice" ICSE Vol. 2, Issue 3, September 2008

[4] Roy K Chanchal,Cordy R James, KoschkeRainer " Comparison and evaluation of code detection techniques and tools" volume 2 issue 24 Febuary, 2009

[5] Kawaguchi Shinji, YamashinayTakanobu, UwanozHidetake, FushidaKyhohei, Kamei Yasutaka , NaguraMasatakaandlidaHajimu "Shinobi: A Tool for AutomaticCode Clone Detection in the IDE" ISSN : 2229-3345 Vol. 4 No. 06 Jun 2013

[6] Nguyen AnhHoan, Nguyen ThanhTung,Pham . H Nam, and Nguyen N Tien"Accurate and Efficient Structural Characteristic Feature Extraction for Clone DetectionVolume 2 issue 8,September 2010.

[7] PuriAmit, "software code clone detection model" Vol. No.1, Issue 3, October 2012

[8] Chanchal K. Roy and James R. Cordy, "An Empirical Study of Function Clones in Open Source Software", 1095-1350/08 $25.00 © 2008 IEEE

[9] Mark Gabel Lingxiao Jiang Zhendong Su, "Scalable Detection of Semantic Clones", ICSE'08, May 10–18, 2008, Leipzig, Germany. Copyright 2008 ACM

[10] Chanchal K. Roy, "Detection and Analysis of Near-Miss Software Clones", 978-1-4244-4828-9/09/$25.00 2009 IEEE

[11] Yoshiki Higo, and Shinji Kusumoto, "Enhancing Quality of Code Clone Detection with Program Dependency Graph", 2009 IEEE

[12] ImanKeivanloo, JuergenRilling, Philippe Charland, "SeClone - A Hybrid Approach to Internet-scale Real-time Code Clone Search", 1063-6897/11 $26.00 © 2011 IEEE

[13] Randy Smith and Susan Harwtiz-detecting and meausingsimilaties in code clones.

[14] Chanchal K. Roy [,a], James R. Cordy[a], Rainer Koschke[b]-Comparison and Evaluation of Code Clone Detection Techniquesand Tools: A Qualitative Approach

[15] K. Kontogiannis, R. DeMori, E. Merlo, M. Galler, and M. Bernstein, Pattern Matching for Clone and Concept Detection, Journal of Automated Software Engineering, 3(1-2):77-108 (1996).